REVIEW OF SPIKING NEURON MODELS FOR NEUROMORPHIC HARDWARE IMPLEMENTATION

[1]XIAOYE WANG, [2]VISHNU P. NAMBIAR, [3]WANG LING GOH, [4]ANH TUAN DO

School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
Institute of Microelectronics,Agency for Science, Technology and Research (A*STAR), Singapore,
E-mail: [1]xiaoye001@e.ntu.edu.sg, [2]ewlgoh@ntu.edu.sg, [3]vishnu paramasivam@ime.a-star.edu.sg, [4]doat@ime.a-star.edu.sg

Abstract - Hardware implementation of spiking neural networks (SNN) has been the focus of many prior works [1-3] due to its relatively low power consumption.When building SNNs, it is typical to consider various forms of neuron models, as it is the key computational unit in the neural network.In this paper, various different digital neuron models will be introduced, followed by an analysis done on each of them, along with comparisons pertaining a series of different features they contain. Three methods of implementing LLIF on hardware will be looked at and comparisons will be made towards their respective components.

Index Terms - Spiking Neuron Model, Spiking Neural Network Hardware, Digital Neuron Circuits.

I. INTRODUCTION

Hardware implementation of spiking neural networks (SNN) has been the focus of many prior works [1-3] due to its relatively low power consumption. The human brain can achieve a very high efficiency in terms of power-performance ratio by consuming a very small amount of power, in the order of a few watts [4]. Custom neuro-hardware chips could possibly achieve this as well, in addition to accelerating neural computation. This is in comparison to what is currently achievable by computing hardware based on traditional Von Neumann architectures. Current research in neuromorphic computing aim to adopt the best power-performance ratios for an efficient SNN hardware implementation.

SNNs are considered as a third-generation neural network [5]. Although SNN may not be mature enough when compared to present popular neural networks, such as CNN and RNN in terms of accuracy, it receives significant attention due to its potential to build a paradigm-shifting computing system with low power consumption [6], low latency [7], and to solve the Von-Neumann bottleneck [8]. In this regard, SNNs very closely mimics biological neural systems.

When building SNNs, it is typical to consider various forms of neuron models, as it is the key computational unit in the neural network. Pre-synaptic neurons and post-synaptic neurons communicate through synapses. Spikes from presynaptic neuron would have either an excitatory or inhibitory influence on postsynaptic neurons, changing its membrane potential values. Once the membrane potential exceeds a certain threshold value, this postsynaptic neuron would then in turn generate a spike which would then propagate to other neurons [9].

In this paper, various different digital neuron models will be introduced, followed by an analysis done on each of them, along with comparisons pertaining a series of different features they contain.Three methods of implementing LLIF on hardware will be looked at andcomparisons will be made towards their respective components.

II. NEURON MODELS

There have been a variety of digital neuron models proposed, specializing in different ways according to different requirements and focuses by researchers. The sections

A. Leaky Integrate-and-Fire (LIF) Neuron Models

Leaky integrate-and-fire model was developed based on the Hodgkin-Huxley model [10], first proposed in 1952 by Alan Lloyd Hodgkin and Andrew Fielding Huxley. It utilized R-C circuits to describe the mechanisms of a neuron's membrane potential.As a model described with a simple analog circuitry,the Hodgkin-Huxley model is widely accepted due to its high accuracy of modeling the real biological plausibility of actual neurons. Neuron features that have been in the Hodgkin-Huxley model include neuron's membrane potential, excitatory and inhibitory synapses and neuron' s membrane decay.

The leaky integrate-and-fire (LIF) model was developed using the Hodgkin-Huxley model as a base. It converts the circuit model into an equationform with various different parameters. The membrane potential of a neuron is tracked through a state variable. Decay rate, threshold and resting voltage are defined using corresponding constants. In the LIF model, membrane potential decay happens in an exponential tendency and is described using a differential equation.The mathematical equation of the LIF model is as below:

$$\tau * \frac{dv}{dt} = v_0 - v + I$$

If $v > \theta$, then fire a spike and $v = v_0$. In the equation, $\tau = RC$, θ is threshold voltage and v_0 is resting voltage. v represents membrane potential and I is the input spike current.This equation form still corresponds to its analog model. In order to use

digital circuits to implement the model, some further conversion need to be done to the equation above.

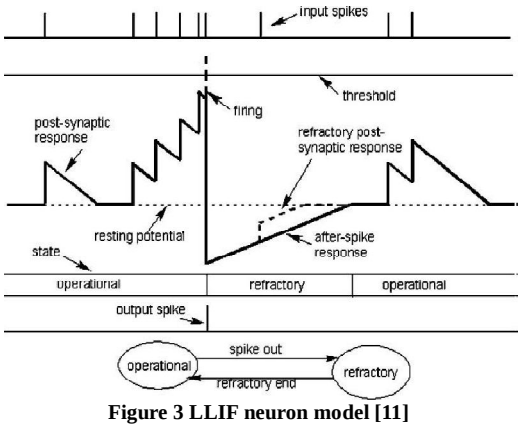


Figure 3 LLIF neuron model [11]

Neuron features described by LIF model contain membrane potential, decay rate, threshold, resting voltage, membrane potential decay form and resting value. Compared to Hodgkin-Huxley model, LIF model is more operational friendly since its computational complexity is lower. Accordingly, it then becomes popular and well accepted as a basic digital neuron model. Recent research works tend to develop new functions or add other features as utilization of the original LIF model.

There are several models developing based on LIF, for example, Linear LIF (LLIF).Being highly accurate can become a disadvantage whencarrying out programs for practical uses. Theoretical computation with high complexity is not convenient to implement on digital hardwarein practice, resulting in low efficiency and a waste of hardware resources. The LLIFmodel is among the most efficient algorithm for hardware implementation, as it is actually a simplified neuronmodel, but still able to perform relatively well.

In LLIF models, the exponential decaying process of membrane potential is substituted with linear decaying [12]. Features related in LLIF models include: membrane potential of neuron, resting potential, threshold, postsynaptic response and after-spike response (membrane potential decay and refractory period). A pulse is employed to represent a spike and a Moore finite state machine is there to help implement the whole model.The figure below illustrates the process of neuron model responding to input spikes.Mathematically, the membrane potentialcan be described as:

$$u(t) = u(t-1) - K\big(u(t-1)\big) + \sum_{i=1}^{n} W_i s_i\,(t-1)$$

$$K\big(u(t)\big) = \begin{cases} k_1 & \text{if } u(t) > U_{rest} \\ -k_2 & \text{otherwise} \end{cases}$$

$u(t)$: membrane potential at time t
U_{rest} : resting potential, constant
n : number of inputs
W_i : weight for input i on the sunapse
$s_i(t)$: input spike at input i, time t
k_1, k_2 : constants which stands for the linear slopes

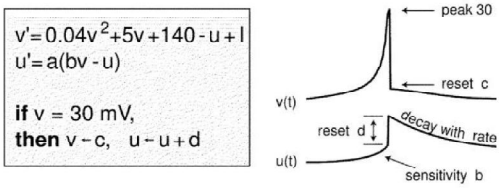


Figure 4Izhikevich neuron model [14]

B. Biologically Inspired Neuron Models

There are a variety of neuronmodels which simplify the Hodgkin-Huxleymodel, and have been implemented in hardware [13-15]. Izhikevichi [15] developed a neuron model taking two factors into consideration: 1) to overcome computational overheads, and 2) to mimic more patterns of real biological neurons.Neuron features considered in Izhikevich model are neuron membrane potential (v) and membrane recovery (u), while four parameters related to these features are adopted to realize neuron behavior of spiking and bursting [14], as listed below.

- Parameter a is utilized to describe how fast is the membrane recovery, and basically a time scale of u.
- Parameter b serves as the scale of the sensitivity of u to the v fluctuation under threshold.
- Parameterc indicatesthe after-spike reset value of v induced by the fast high-threshold K^+ conductance, with a typical value of -65 mV.
- Parameter d controls the after-spike reset of u with a typical value of 2.

Spiking patterns would vary according to different choices of these four parameters. The flexibility on parameter choosing improves the mimicking ability of the neuron model, while keeping the complexity in an acceptable degree.

C. Other Neuron Models

Other neuron models drift further towards the conventional artificial neural network designs, in favor of pattern recognition tasks and applications. TheMcCulloch-Pitts model [16] falls in this category, with variations on the activation function implementation such as basic sigmoid function [17], hyperbolic tangent function [18] and piecewise linearfunction[19].

III. COMPARISON OF NEURON MODELS

Evaluation towards a certain neuron model would have different criterion under different circumstances. This part of review and comparison only focus on a few general important points. Firstly, number and kind of neuron features that each model has. Secondly, whether and to what kind of degree themodels are friendly to hardware implementation. Thirdly, how flexible and dynamic the neuron models can be.

LIF is the one that has the highest similarity with Hodgkin-Huxley model, adopting exponential decay. Linear LIF model decaying mode is in linear form and

it isfriendlier in terms of hardware implementation.Izhikevich neuron model considers both computational simplicity and plausibility, so its complexity degree should lie in district between the former two.

Table 1 Neuron features

Neuron Features	LIF	LLIF	Izhikevich Model
Membrane Potential	√	√	√
Threshold Potential	√	√	
Resting Potential	√	√	
Potential Decay	√	√	
Postsynaptic Response		√	
Refractory Period		√	
Membrane Recovery			√

LIF model and LLIF model are only flexible on the decision of constants in their respective membrane potential decaying form. In contrast, Izhikevich displays much more dynamics due to its four parameters which can be customized according to different needs and preferences.

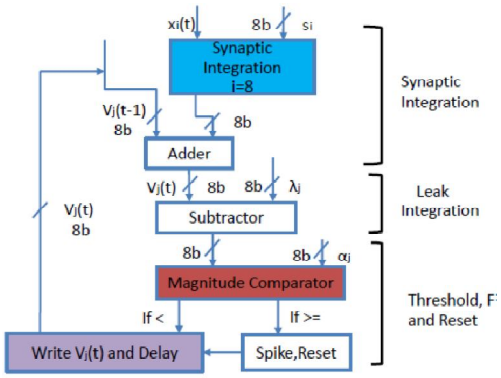


Figure 5Basic LIF neuron implementation [20];components include adder, subtractor, comparator, reset register and memory register.

IV. IMPLEMENTATIONCOMPARISON

Through comparison of neuron models in the last section, the basic features and advantages of the three models are generally clear.Izhikevich neuron model is relatively too complicated to implement on hardware. More instances at present also focus on implementing LIF on digital hardware circuits. In this section, several ways of implementing the LLIFneuron model on hardware are discussed.

A. Basic LIF neuron design

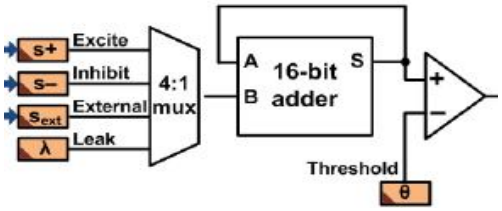


Figure 6 Digital CMOS neuron from IBM [21]; components include adder, mux and comparator

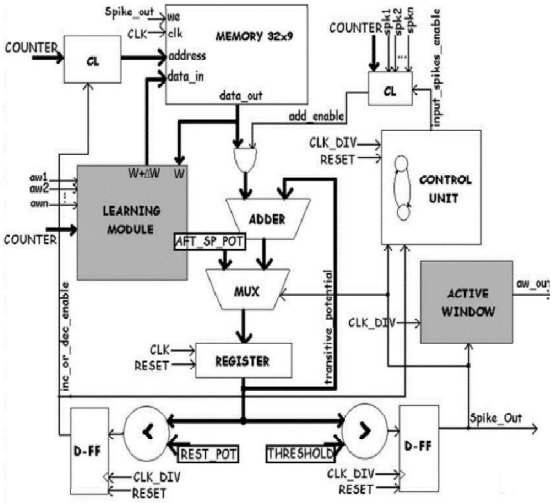


Figure 7LLIF neuron implementaion[11];components includes adder, mux, comparator, reset register, memory register and finite state machine.

In this very basic LIF neuron model implementation on hardware, there are roughly three parts. The first part is the synaptic integration which contains an adder circuit that accumulates response of input spikes. A subtractor is put in the leak integration part to provide a constant leakage value to the membrane potential. In Magnitude Comparator, the value of membrane potential will be compared to a specific threshold and do corresponding operation (fire a spike and reset, or directly write to memory) according to the comparing results.

B. CMOS neuron design from IBMTrueNorth

The implementation is part of the neuromorphic chip from IBM. The whole architecture is clearly described in the publication. A mux, a 16-bit adder and a comparator. It integrates the accumulation of input responses and leak value together via a mux. The design implements both excitatory and inhibitory spikes, and also external input value. In this way, the area needed for implementation would be less.

C. LLIF neuron design

This model achieved linear leakage base on original LIF model. The model has a control unit which is a finite state machine. With the state machine, the neuron state can change between operational mode and refractory mode. It is also a design that achieved on-chip learning with learning modules.

D. Comparison Summary

From the descriptions of these various different neuron hardware implementations,the differences can be tabulated for comparison, as shown in Table 2. Using the basic LIF neuron implementation as a reference, the IBM neuron model integrate adder and subtractor through a mux, decreasing the area needed for layout. LLIF neuron model is relatively more complex and contains more operation components. However, its function is more complete and the

ability to do on-chip learning is one of its great advantages.

Table 1 Hardware implementation comparison

	LIF[20]	IBM Digital Neuron[21]	LLIF[11]
Adder	√	√	√
Subtractor	√		
Mux		√	√
Comparator	√	√	√
Reset register	√		√
Memory register	√		√
FSM			√
Area needed	Medium	Small	Large

CONCLUSION

In this paper, various different kinds of digital neuron models are introduced respectively, specifically focusing on the LIF models. Also, three different ways to implement the digital neuron model on hardware are discussed pertaining area and hardware resources needed.

All the three models have their own advantage and relative deficiency. LIF Model is the one that is the most biologically plausibly, while LLIF Model describes the largest number of neuron features., which partly explains why LLIF is commonly used by researchers.As for hardware implementation, theLLIF neuron model ticks more boxes when it comes to function, sacrificing the hardware area and resources due to the required hardware components for this.

REFERENCES

[1] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, et.al, "A million spiking-neuron integrated circuit with a scalable communication network and interface," Science, vol. 345, no. 6197, pp. 668–673, 2014.

[2] N. Qiao, H. Mostafa, F. Corradi, M. Osswald, F. Stefanini, D. Sumislawska and G. Indiveri, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128K synapses," Front. Neurosci., vol. 9, no. APR, pp. 1–17, 2015.

[3] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, and Others, "Loihi: A neuromorphic manycore processor with on-chip learning," IEEE Micro, vol. 38, no. 1, pp. 82–99, 2018.

[4] C. Mead, "Neuromorphic electronic systems," Proceedings of the IEEE, vol. 78, no. 10, pp. 1629-1636, 1990.

[5] W. Maass, "Networks of spiking neurons: The third generation of neural network models, Neural Networks, vol.10, pp. 1659-1671, 1997.

[6] P. Leong and M. Jabri, "A VLSI neural network for morphology classification," International Joint Conference on Neural Networks (IJCNN), 1992, vol. 2, pp. 678–683.

[7] P. Hasler and L. Akers, "VLSI neural systems and circuits," IEEE 9th Annual International Phoenix Conference on Computers and Communications, 1990, pp. 31-37.

[8] J. Backus, "Can programming be liberated from the von neumann style?" A functional style and its algebra of programs," Commun. ACM, vol. 21, no. 8, pp. 613–641, Aug. 1978.

[9] R. Douglas, M. Mahowald, and C. Mead, "Neuromorphic analogue vlsi," Annual Review of Neuroscience, vol. 18, no. 1, pp. 255–281, 1995

[10] L. Hodgkin and A. F. Huxley, "A Quantitative Description of Membrane Current and Its Application to Conduction and Excitation in Nerve," The J. of Physiology, vol. 117, 1952.

[11] A. Upegui, C. A. Pena-Reyes and E. Sanchez, "An FPGA platform for on-line topology exploration of spiking neural networks," Microprocessors and Microsystems, 2005.

[12] R. B. Stein, "A Theoretical Analysis of Neuronal Variability," Biophysical J., vol. 5, 1965.

[13] S. Binczak, S. Jacquir, J.-M. Bilbault, V. B. Kazantsev, and V. I. Nekorkin, "Experimental study of electrical fitzhugh–nagumo neurons with modified excitability," Neural Networks, vol. 19, no. 5, pp. 684–693, 2006.

[14] M. Hayati, M. Nouri, D. Abbott, and S. Haghiri, "Digital multiplierless realization of two-coupled biological hindmarsh–rose neuron model," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 63, no. 5, pp. 463–467, 2016.

[15] E. M. Izhikevich, "Simple model of spiking neurons," IEEE Transactions on Neural Networks, vol. 14, no. 6, pp. 1569-1572, 2003.

[16] W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," The bulletin of mathematical biophysics, vol. 5, no. 4, pp. 115–133, 1943.

[17] S. Al-Kazzaz and R. Khalil, "Fpga implementation of artificial neurons: Comparison study," in Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on. IEEE, 2008, pp. 1–6.

[18] D. Baptista and F. Morgado-Dias, "Low-resource hardware implementation of the hyperbolic tangent for artificial neural networks," Neural Computing and Applications, vol. 23, no. 3-4, pp. 601–607, 2013.

[19] H. Hikawa, "A digital hardware pulse-mode neuron with piecewise linear activation function," Neural Networks, IEEE Transactions on,vol. 14, no. 5, pp. 1028–1037, 2003.

[20] K. Suzuki, Y. Okuyama, A. B. Abdallah, "Hardware Design of a Leaky Integrate and Fire Neuron Core Towards the Design of a Low-power Neuro-inspired Spike-based Multicore SoC," Information Processing Society, Tohoku Branch Conference, 2018.

[21] Seo, J. s, B. Brezzo, Y. Liu, B. D. Parker, S. K. Esser, R. K. Montoye, B. Rajendran, et al. "A 45nm CMOS Neuromorphic Chip with a Scalable Architecture for Learning in Networks of Spiking Neurons". In 2011 IEEE Custom Integrated Circuits Conference (CICC), 1–4, 2011.

[22] This research is supported by Programmatic grant no. A1687b0033 from the Singapore government's Research, Innovation and Enterprise 2020 plan (Advanced Manufacturing and Engineering domain)

★★★